National University
of computer and emerging sciences

Assignment 1

CS-326 Parallel and Distributed Computing
BS(CS) – D
Batch 2018

Submitted By:

Hassan Shahzad 18i-0441

Submitted to:

Sir Ehtisham Zahoor

Date of Submission:

16-04-21

Explanation:

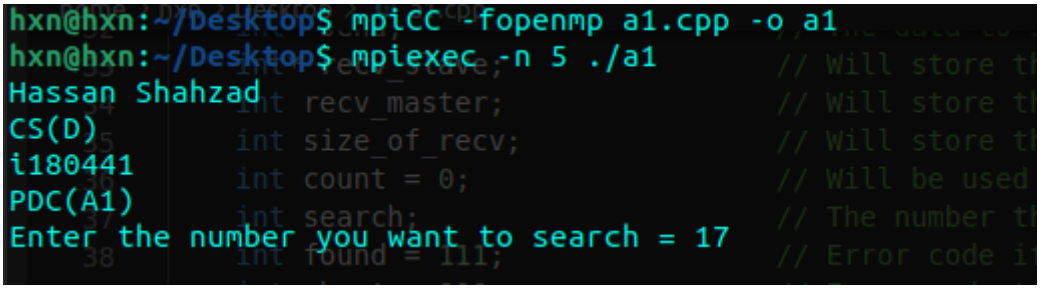
My implementation taken of the assignment works with the following logic. Number of processes will be passed as an argument. From those numbers, 1 process (rank =0) will be the **master process** whereas the rest of the processes will be **slave processes**. Then the master process will distribute the data and the slave processes will each perform search simultaneously and one a process has found the number; each process will be terminated and master will display the message.

In this implementation, I am using both **OpenMP** and **MPI**. OpenMP is used on multiple occasions especially for **synchronization**.

Furthermore, stepwise implementation of the assignment is listed below:

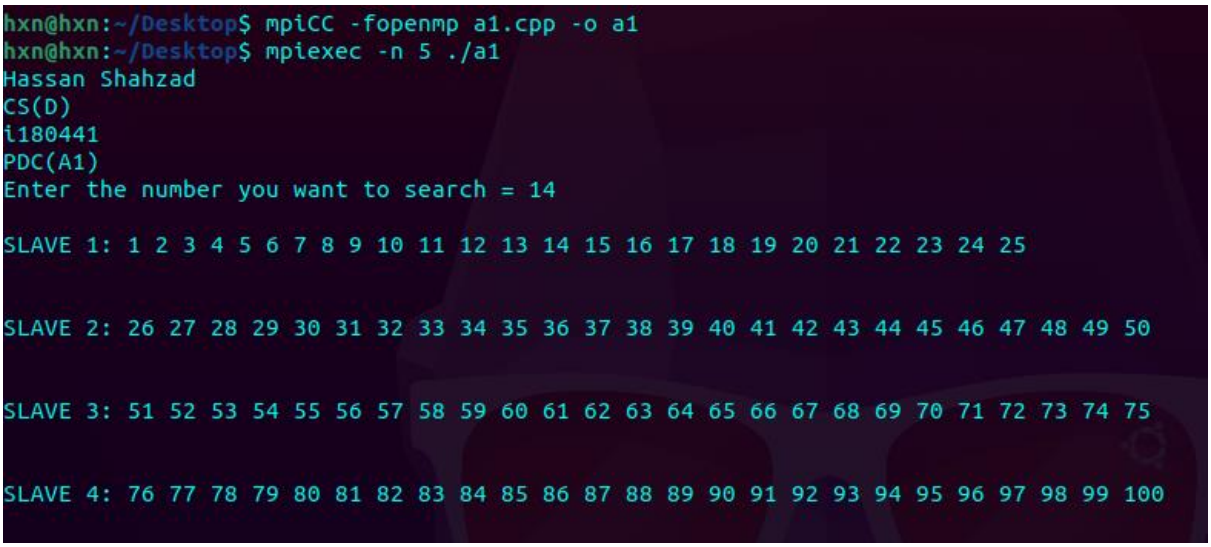
1. Step 1:

First of all, I asked the user to **input** the number he/she wanted to search.



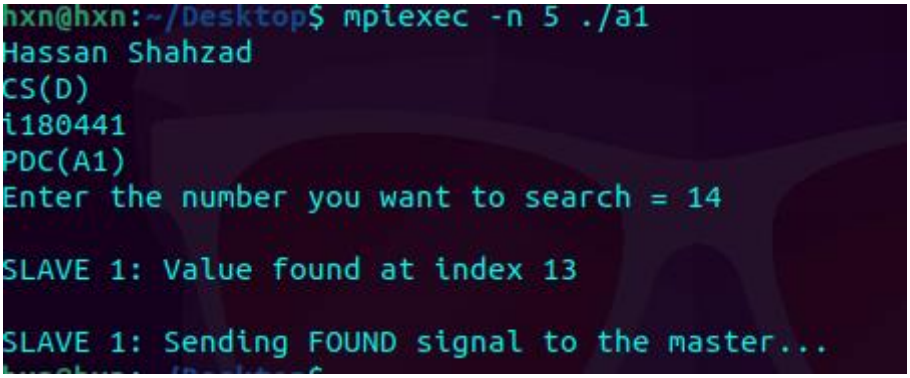
2. Step 2:

The master process then **divides** the data into **equal parts** and **distributes** it among the slave processes. And the slave processes then each displays the local data they were given.



3. Step 3:

In this step, each of the slave process **searches** for the value entered in their allocated chunks. When one slave process finds the number, it sends a **signal** to the master process along with displaying the index at which the data was found.

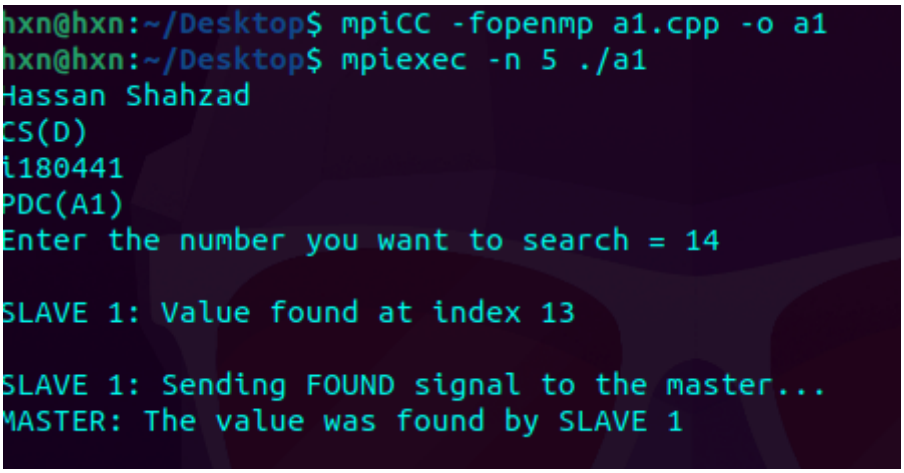


4. Step 4:

In this step, the master node receives the signal and verifies the signal code with the pre-defined signal codes. I have kept **two pre-defined signal codes** which are as follows:

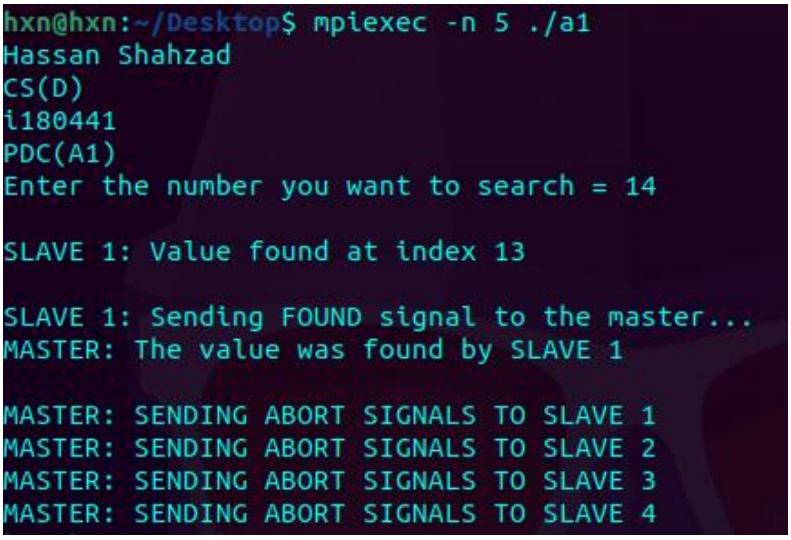
- *Found = 111*
- *Abort = 000*

After verifying, the master process then generates a message acknowledging which particular slave found the number.



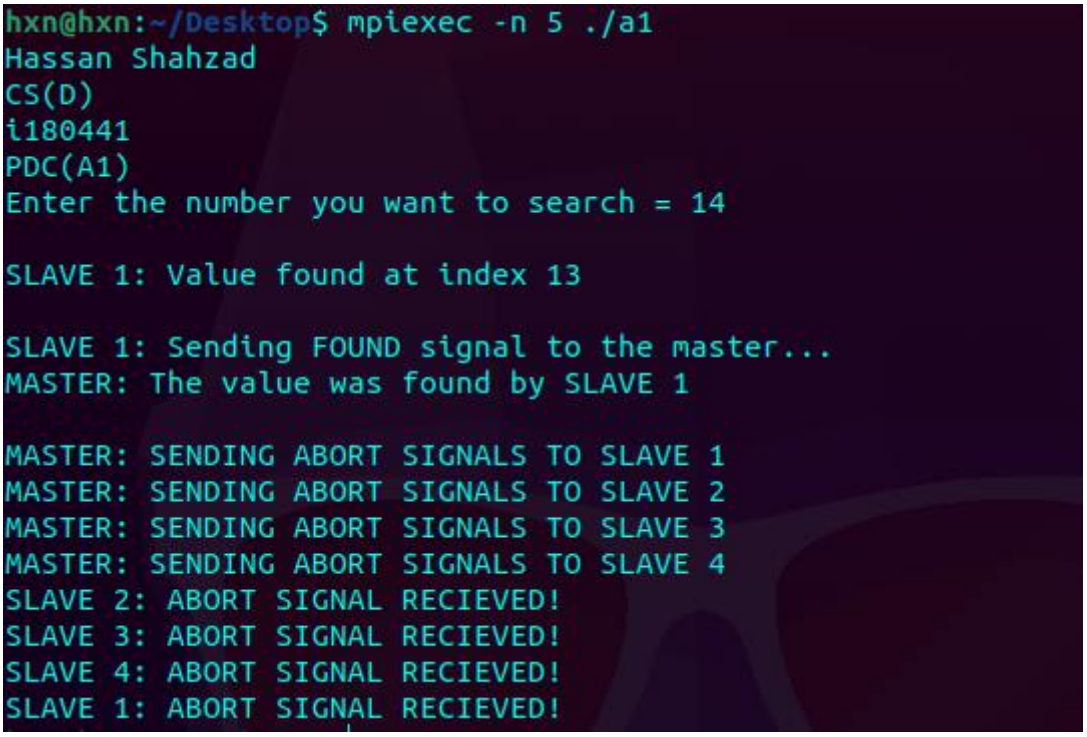
5. Step 5:

Once the master verifies the found signal, it sends an ***abort message*** to each and every slave asking them to abort their search.



6. Step 6:

Each slave receives the abort signal after which it aborts and displays the following message.



7. Step 7:

This step is an ERROR CONDITION. I put in some checks such that if a user wants to search a number that doesn't exist in a dataset, each process will display "VALUE NOT FOUND" message and exit.

```
hxn@hxn:~/Desktop$ mpiexec -n 5 ./a1
Hassan Shahzad
CS(D)
i180441
PDC(A1)
Enter the number you want to search = 510

SLAVE 3: VALUE NOT FOUND!
SLAVE 1: VALUE NOT FOUND!
SLAVE 2: VALUE NOT FOUND!
SLAVE 4: VALUE NOT FOUND!
```

8. Examples:

Following are some example outputs of the program with various inputs:

```
Hassan Shahzad
CS(D)
i180441
PDC(A1)
Enter the number you want to search = 14

SLAVE 2: 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50


SLAVE 4: 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100


SLAVE 3: 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75


SLAVE 1: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25


SLAVE 1: Value found at index 13

SLAVE 1: Sending FOUND signal to the master...
MASTER: The value was found by SLAVE 1

MASTER: SENDING ABORT SIGNALS TO SLAVE 1
MASTER: SENDING ABORT SIGNALS TO SLAVE 2
MASTER: SENDING ABORT SIGNALS TO SLAVE 3
MASTER: SENDING ABORT SIGNALS TO SLAVE 4
SLAVE 1: ABORT SIGNAL RECIEVED!
SLAVE 3: ABORT SIGNAL RECIEVED!
SLAVE 4: ABORT SIGNAL RECIEVED!
SLAVE 2: ABORT SIGNAL RECIEVED!
```

Fig 8.1: *When value "14" is entered.*

```
hxn@hxn:~/Desktop$ mpiexec -n 5 ./a1
Hassan Shahzad
CS(D)
i180441
PDC(A1)
Enter the number you want to search = 59

SLAVE 2: 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50


SLAVE 3: 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75


SLAVE 1: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25


SLAVE 4: 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100


SLAVE 3: Value found at index 8

SLAVE 3: Sending FOUND signal to the master...
MASTER: The value was found by SLAVE 3

MASTER: SENDING ABORT SIGNALS TO SLAVE 1
MASTER: SENDING ABORT SIGNALS TO SLAVE 2
MASTER: SENDING ABORT SIGNALS TO SLAVE 3
MASTER: SENDING ABORT SIGNALS TO SLAVE 4
SLAVE 2: ABORT SIGNAL RECIEVED!
SLAVE 3: ABORT SIGNAL RECIEVED!
SLAVE 1: ABORT SIGNAL RECIEVED!
SLAVE 4: ABORT SIGNAL RECIEVED!
```

Fig 8.2: *When value "59" is entered.*

```
hxn@hxn:~/Desktop$ mpiexec -n 5 ./a1
Hassan Shahzad
CS(D)
i180441
PDC(A1)
Enter the number you want to search = 99

SLAVE 1: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25


SLAVE 2: 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50


SLAVE 4: 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100


SLAVE 3: 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75


SLAVE 4: Value found at index 23

SLAVE 4: Sending FOUND signal to the master...
MASTER: The value was found by SLAVE 4

MASTER: SENDING ABORT SIGNALS TO SLAVE 1
MASTER: SENDING ABORT SIGNALS TO SLAVE 2
MASTER: SENDING ABORT SIGNALS TO SLAVE 3
MASTER: SENDING ABORT SIGNALS TO SLAVE 4
SLAVE 1: ABORT SIGNAL RECIEVED!
SLAVE 3: ABORT SIGNAL RECIEVED!
SLAVE 4: ABORT SIGNAL RECIEVED!
SLAVE 2: ABORT SIGNAL RECIEVED!
```

Fig 8.3: *When value "99" is entered.*

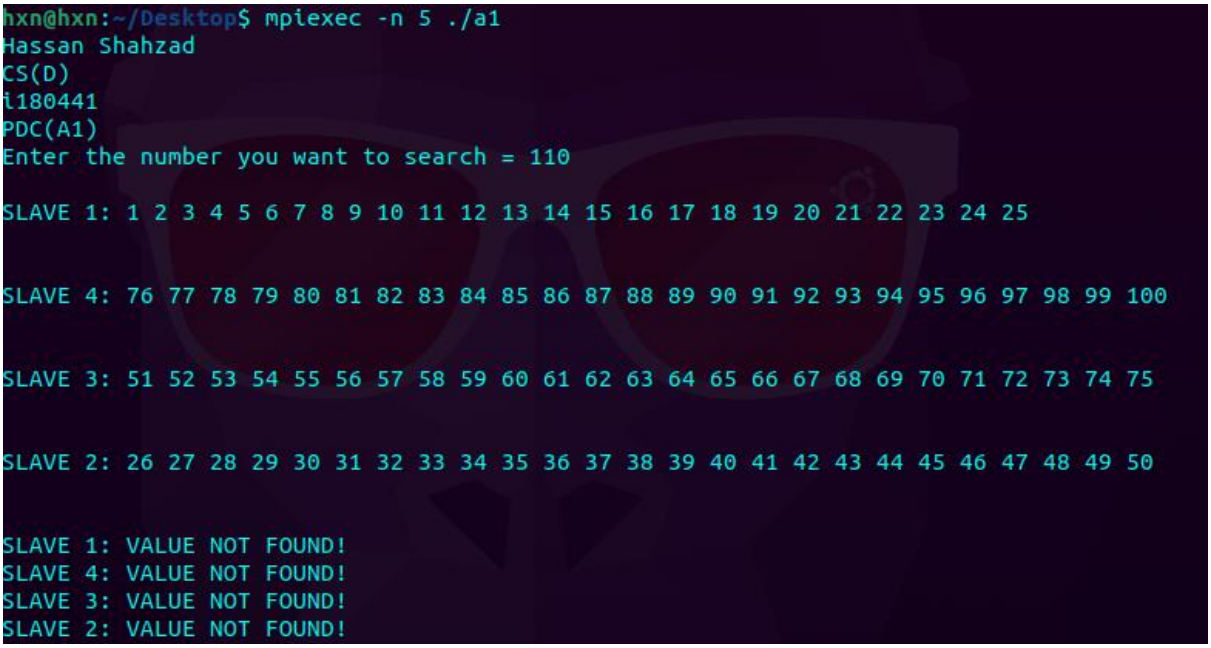


Fig 8.4: *When value "110" (not in dataset) is entered.*

Requirements Checklist:

☒ Program is written in C/C++
☒ You must use both OpenMP and MPI
☒ Program is executing without any warnings
☒ Screenshots of outputs are attached
☒ A single zip file uploaded
☒ List is equally distributed
☒ Proper searching and aborting functionality
☒ Proper commenting and detailed screenshot (+additional explanation)

In the above checklist, **"x"** represents **"completed"**. From the above analysis, we can conclude that all of the requirements were properly implemented and fulfilled. 😊